Theoretical Computer Science 31 (1984) 41–48
North-Holland

MINIMAL PAIRS FOR P

Uwe SCHÖNING*
Institut für Informatik, Universität Stuttgart, D-7000 Stuttgart 1, West Germany

Communicated by R. Book
Received April 1983

Abstract. We prove a general minimal pair theorem which yields as corollaries many results about minimal pairs for P obtained previously by various authors. Furthermore, it yields the new counterintuitive result that there exist 'arbitrarily complex' minimal pairs. We also investigate the question of whether minimal pairs in NP are 'low'.

1. Introduction

The study of minimal pairs in recursive function theory was motivated by providing a refutation of Shoenfield's homogeneity conjecture for the r.e. degrees (cf. [14]). It is known that there exist r.e. minimal pairs. Inspired by the recursion theoretic results, many authors investigated the question of whether minimal pairs for P exist. Ladner [8] showed that minimal pairs for P exist, but he gave no upper bound on the complexity of the minimal pair. Machtey [10] proved that minimal pairs for P exist with sub-exponential complexity. Breidbart [3] and Landweber et al. [9] show the existence of minimal pairs 'below' each non-polynomial set. Thus, there exist minimal pairs 'arbitrarily close' to P. A consequence is that if $P \neq NP$, then there exist minimal pairs in $NP - P$. This could be compared with the recursion theoretic result by Lachlan [6] that there are r.e. degrees below which there are no minimal pairs.

Chew and Machtey [4] give an elegant treatment of complexity theoretic density and minimal pair results which is intended to give the general idea of the 'delayed diagonalization' and 'looking back' proof techniques rather than to obtain the strongest possible results.

In this paper we present a very general minimal pair theorem which is formulated in a similar fashion to that of the uniform diagonalization theorem in [12]. Virtually all former results about minimal pairs can be obtained as corollaries of our theorem. We believe that a theorem like the one presented here, together with a clean detailed proof, though being more technical than in [4], can contribute to the clarification

* This research was supported in part by Deutsche Forschungsgemeinschaft and by the National Science Foundation under grant MCS80-11979, and was carried out while the author was visiting the Department of Mathematics, University of California, Santa Barbara, CA 93106, U.S.A.

of more complex proof techniques. Some further corollaries of the theorem show that there exist minimal pairs for P of arbitrary complexity, which seems counter-intuitive. Further, we investigate the question of whether minimal pairs in NP are 'low'.

2. Preliminaries

All our sets are languages over some fixed alphabet Σ, $|\Sigma| \geq 2$. For a string $w \in \Sigma^*$, $|w|$ denotes the length of w. We assume a total ordering of Σ^* such that shorter strings precede longer ones and strings of the same length are ordered lexicographically, $\Sigma^* = \{w_0, w_1, w_2, \ldots\}$.

Our model of computation is the multitape Turing machine. For a Turing machine M, let $L(M)$ denote the set accepted by M. A (possibly nondeterministic) Turing machine M is $f(n)$ time bounded for a function $f: N \to N$ if for all inputs of length n, M makes at most $f(n)$ steps on all computation paths. If f can be taken to be a polynomial, then M is polynomial time bounded. Define

$$\text{DTIME}(f(n)) = \{A \subseteq \Sigma^* \mid A = L(M) \text{ for some deterministic } f(n) \text{ time bounded Turing machine } M\},$$

$$\text{NTIME}(f(n)) = \{A \subseteq \Sigma^* \mid A = L(M) \text{ for some nondeterministic } f(n) \text{ time bounded Turing machine } M\},$$

$$\text{P} = \bigcup_{k>0} \text{DTIME}(n^k), \qquad \text{NP} = \bigcup_{k>0} \text{NTIME}(n^k),$$

$$\text{DEXPTIME} = \bigcup_{c>0} \text{DTIME}(2^{cn}), \qquad \text{NEXPTIME} = \bigcup_{c>0} \text{NTIME}(2^{cn}).$$

We will consider the following polynomial time reducibilities. Define $A \leq_m^P B$ if and only if there exists a function $f: \Sigma^* \to \Sigma^*$ that can be computed by a deterministic, polynomial time bounded Turing machine such that for each $x \in \Sigma^*$, $x \in A \Leftrightarrow f(x) \in B$. Define $A \leq_1^P B$ if and only if there exists a deterministic, polynomial time oracle Turing machine M such that for each $x \in \Sigma^*$, $x \in A \Leftrightarrow M$ accepts x when using oracle B. We write $A <_m^P B$ ($A <_1^P B$) if $A \leq_m^P B$ ($A \leq_1^P B$) and not $B \leq_m^P A$ (not $B \leq_1^P A$, respectively).

A function $t: N \to N$ is *time constructible* if there exists a deterministic Turing machine which on input of length n halts after exactly $t(n)$ steps.

Recall that each total recursive function can be majorized by some time constructible function (see [12]).

Let $A \triangle B$ denote the symmetric difference of A and B, $A \triangle B = (A - B) \cup (B - A)$. A class of sets $\mathcal{C}$ is *closed under finite variations* if for all sets A, B, $A \in \mathcal{C}$ and $A \triangle B$ is finite implies $B \in \mathcal{C}$. $\mathcal{C}$ is *recursively presentable* if there exists a recursively enumerable sequence of Turing machines $P_0, P_1, P_2, \ldots$ such that each P_i halts on all inputs and $\mathcal{C} = \{L(P_i) \mid i \geq 0\}$.

Observe that this definition is not quite standard. Each recursively presentable class $\mathcal{C}$ consists only of recursive sets. In fact, if $\mathcal{C}$ is recursively presentable where

$P_0, P_1, P_2, \ldots$ is the recursive presentation, then it is easy to obtain a recursive set A that is not in $\mathscr{C}$: define $w_n \in A$ if and only if $w_n \notin L(P_n)$.

In [9, 12] a certain 'gap language' $G[r]$ is introduced. Here we will consider a slightly more sophisticated version. For a function $r: N \to N$, $r(n) > n$, and natural numbers i, n, $0 \leq i < n$, define

$$G_i^n[r] = \{x \in \Sigma^* \mid r^{(k)}(0) \leq |x| < r^{(k+1)}(0) \text{ for some } k,\ k = i \pmod n\}.$$

Here $r^{(k)}(\ )$ means k-fold application of the function r. Observe that $G_0^2[r]$ coincides with the earlier definition of $G[r]$ in [9, 12]. By an argument similar to that in [12], for each i and n, $0 \leq i \leq n$, and time constructible function r, $r(n) > n$, the set $G_i^n[r]$ is in P.

A pair of sets $\{B_1, B_2\}$ is called a *minimal pair* (for P) if and only if

(a) $B_i \notin \mathrm{P}$, $i = 1, 2$;

(b) $D \leq_1^{\mathrm{P}} B_i$, $i = 1, 2$ implies $D \in \mathrm{P}$.

It is possible to generalize this definition to other subrecursive reducibilities. In this case, every occurrence of 'P' in the above definition must be replaced by the 0-degree of the reducibility under consideration. The following theorem is formulated in terms of the above minimal pair definition which fixes $\leq_1^{\mathrm{P}}$-reducibility and P (the 0-degree of $\leq_1^{\mathrm{P}}$), but it is not hard to expand the theorem and its proof to such other subrecursive reducibilities.

3. The minimal pair theorem

In a similar fashion to that of the uniform diagonalization theorem in [12], we prove a very general theorem that allows us to prove the existence of minimal pairs in various complexity classes.

Theorem 1. *Let A_1, A_2 be recursive sets and let $\mathscr{C} \supseteq \mathrm{P}$ be a class of sets such that the following hold:*

(a) $A_1 \notin \mathscr{C}$, $A_2 \notin \mathscr{C}$,

(b) $\mathscr{C}$ *is recursively presentable,*

(c) $\mathscr{C}$ *is closed under finite variations.*

Then there exist recursive sets B_1, B_2 such that

(d) $B_1 \notin \mathscr{C}$, $B_2 \notin \mathscr{C}$,

(e) $B_1 \leq_{\mathrm{m}}^{\mathrm{P}} A_1$, $B_2 \leq_{\mathrm{m}}^{\mathrm{P}} A_2$,

(f) $\{B_1, B_2\}$ *is a minimal pair.*

Proof. Let $P_0, P_1, P_2, \ldots$ be the recursive presentation of $\mathscr{C}$. Define functions $r_1, r_2: N \to N$ as follows:

$$r_1(n) = 1 + \max\{\text{the length of the least } z \text{ such that } |z| \leq n \text{ and } z \in A_1 \triangle L(P_i)\}.$$

and

$$r_2(n) = 1 + \max_{i<n} \{\text{the length of the least } z \text{ such that } |z| \geq n \text{ and } z \in A_2 \triangle L(P_i)\}.$$

Since $A_1 \notin \mathscr{C}$ and $\mathscr{C}$ is closed under finite variations, for each i, there exist infinitely many z such that $z \in A_1 \triangle L(P_i)$. Therefore, and by the fact that each P_i halts on each input, r_1 is a total recursive function. The same argument applies to r_2.

Since A_1 and A_2 are recursive, there exist Turing machines M_1 and M_2 which halt on each input such that $A_1 = L(M_1)$ and $A_2 = L(M_2)$. Let $t_1, t_2 : N \to N$ be monotone increasing recursive functions that bound the number of steps of M_1 and M_2 on inputs of length n, respectively.

Let $s(n) \geq \max\{r_1(n), r_2(n), t_1(n), t_2(n), 2^n\}$ be a time-constructible function. Then for each i and n, $0 \leq i < n$, $G_i^n[s] \in \mathrm{P}$. Define $B_1 = A_1 \cap G_0^n[s]$ and $B_2 = A_2 \cap G_3^n[s]$. We verify that B_1, B_2 have the desired properties.

First, suppose $B_1 \in \mathscr{C}$, then there exists a j such that $B = L(P_j)$. Choose a k such that $s^{(k)}(0) \geq j$ and $k = 0 \pmod 6$. By the construction of r_1, there exists a string z, $s^{(k)}(0) \leq |z| < s^{(k+1)}(0)$, such that $z \in A_1 \triangle L(P_j)$. Since A_1 and B_1 are identical on the set $\{z \mid s^{(k)}(0) \leq |z| < s^{(k+1)}(0)\} \subseteq G_0^6[s]$, we have $z \in B_1 \triangle L(P_j)$, a contradiction, which proves that $B_1 \notin \mathscr{C}$. A similar argument shows that $B_2 \notin \mathscr{C}$.

Second, the following polynomial-time computable function f witnesses that $B_1 \leq_{\mathrm{m}}^{\mathrm{P}} A_1$:

$$f(x) = \begin{cases} x & \text{if } x \in G_0^6[s], \\ x_0 & \text{otherwise,} \end{cases}$$

where x_0 is a fixed element not in A_1. (Observe that $\bar{A}_1 \neq 0$ because $A_1 \notin \mathrm{P}$.) A similar argument proves that $B_2 \leq_{\mathrm{m}}^{\mathrm{P}} A_2$.

Third, suppose for some set F, $F \leq_1^{\mathrm{P}} B_1 = A_1 \cap G_0^6[s]$ and $F \leq_1^{\mathrm{P}} B_2 = A_2 \cap G_3^6[s]$. Let Q_1, Q_2 be deterministic, polynomial-time bounded oracle machines that witness the above reductions. It can be easily seen that the following Turing machine accepts F:

On input x:
if $x \in G_2^6[s] \cup G_3^6[s] \cup G_4^6[s]$
  then simulate Q_1 on input x; whenever Q_1 goes into its query state and y is the string on the query tape:
    if $y \notin G_0^6[s]$
      then continue in the no-state of Q_1
      else continue in the yes- (no-) state of Q_1 if M_1 accepts (rejects) input y.
    Accept if Q_1 accepts; else reject.
  else simulate Q_2 on input x; whenever Q_2 goes into its query state and y is the string on the query tape:
    if $y \notin G_3^6[s]$
      then continue in the no-state of Q_2

else continue in the yes- (no-) state of Q_2 if M_2 accepts (rejects) input y.
Accept if Q_2 accepts; else reject.

We claim that this algorithm is polynomial time-bounded. Clearly, all the case decisions can be made in polynomial time. Suppose that the first **then**-case occurs, that is, $x \in G_2^6[s] \cup G_3^6[s] \cup G_4^6[s]$. Then for some k, $k = 2 \pmod 6$, $s^{(k)}(0) \leq |x| < s^{(k+3)}(0)$. Let y be a string that appears on Q_1's query tape in a computation with input x. Then for almost all x, $|y| < 2^{|x|} \leq 2^{s^{(k+3)}(0)} \leq s^{(k+4)}(0)$ because $s(n) \geq 2^n$. It follows that if $y \in G_0^6[s]$, then $|y| \leq s^{(k-1)}(0)$. Hence the run time of M_1 on y is at most $t_1(|y|) \leq t_1(s^{(k-1)}(0)) \leq s^{(k)}(0) \leq |x|$ because t_1 is monotone increasing and $s(n) \geq t_1(n)$. Hence, in this case, $y \in ? L(M_1) = A_1$ can be decided in at most $|x|$ steps.

Since the argument is completely symmetric if $x \notin G_2^6[s] \cup G_3^6[s] \cup G_4^6[s]$, we have shown that $F \in \mathrm{P}$. □

4. Applications

We now proceed with some applications of the Minimal Pair Theorem.

Corollary 2 ([9]). *If* $\mathrm{P} \neq \mathrm{NP}$, *then there exist minimal pairs in* $\mathrm{NP} - \mathrm{P}$.

Proof. Choose any NP-complete set (e.g., SAT) for A_1 and A_2 in the theorem and $\mathcal{C} = \mathrm{P}$. P is clearly closed under finite variations and recursively presentable (see [12]). Then the theorem yields a minimal pair in $\mathrm{NP} - \mathrm{P}$. □

Observe that the minimal pair in Corollary 2 is $\leq_1^{\mathrm{P}}$-incomparable and thus not $\leq_1^{\mathrm{P}}$-complete for NP.

It is also possible to use the theorem to construct minimal pairs in two different complexity classes.

Corollary 3. *If* $\mathrm{NP} \neq \mathrm{co\text{-}NP}$, *then there exists a minimal pair* $\{B_1, B_2\}$ *such that* $B_1 \in \mathrm{NP\text{–}co\text{-}NP}$, *and* $B_2 \in \mathrm{co\text{-}NP\text{–}NP}$.

Proof. Choose in the theorem $A_1 = \mathrm{SAT}$, $A_2 = \overline{\mathrm{SAT}}$ and $\mathcal{C} = \mathrm{NP} \cap \mathrm{co\text{-}NP}$. A recursive presentation of $\mathrm{NP} \cap \mathrm{co\text{-}NP}$ can be obtained by merging the recursive presentations of NP and co-NP (cf. [12]). □

The following corollary is similar to theorems appearing in [9] and [3].

Corollary 4. *Let* X, Y *be recursive sets such that* $X <_1^{\mathrm{P}} Y$. *Then there exist recursive sets* D_1, D_2 *such that*

(a) $X \leq_{\mathrm{m}}^{\mathrm{P}} D_i \leq_1^{\mathrm{P}} Y$, $i = 1, 2$.

(b) $Y \not\leq_T^P D_i$, $D_i \not\leq_T^P X$, $i = 1, 2$,
(c) $F \leq_T^P D_i$, $i = 1, 2$ *implies* $F \leq_T^P X$.

Proof. Apply the Minimal Pair Theorem with $\mathscr{C} = \{E \mid E \leq_T^P X\}$ and $A_1 = A_2 = Y$. Then $\mathscr{C}$ is clearly recursively presentable by combining a recursive enumeration of all polynomial-time and deterministic oracle machines with a decision procedure for X (see [3]). The theorem provides a minimal pair $\{B_1, B_2\}$ such that $B_i \notin \mathscr{C}$, $i = 1, 2$, that is, $B_i \not\leq_T^P X$. Now define $D_i = 0B_i \cup 1X$, $i = 1, 2$. It follows that $X \leq_m^P D_i \leq_T^P Y$, $i = 1, 2$, and $Y \not\leq_T^P D_i$, $D_i \not\leq_T^P X$. Let F be a set such that $F \leq_T^P D_i$, $i = 1, 2$. Then the polynomial-time procedure for F in the proof of the Minimal Pair Theorem becomes a polynomial-time procedure relative to X, hence $F \leq_T^P X$. □

Observe that the results by Breidbart [3] and Landweber et al. [9] mentioned in Section 1 are special cases of Corollary 4. Using recursion theoretic terminology, Corollary 4 asserts that all polynomial Turing degrees are branching. This should be contrasted with Lachlan's result [7] that there are non-branching r.e. degrees.

The next corollaries are somewhat counterintuitive because they state that in some sense there exist 'arbitrarily complex' minimal pairs.

Corollary 5. *Let $\mathscr{C} \supseteq \mathrm{P}$ be any recursively presentable class that is closed under finite variations. Let A be a recursive set which is not in $\mathscr{C}$. Then there exist recursive sets $B_1, B_2 \leq_m^P A$ which are not in $\mathscr{C}$ and are minimal pairs.*

Proof. Apply the Minimal Pair Theorem with $A_1 = A_2 = A$. □

Next we show that it is possible to obtain minimal pairs satisfying any given upper or lower time complexity bound.

Let a function $t: N \to N$ be *super-polynomial* if t eventually majorizes each polynomial function.

Corollary 6. *Let t_1, t_2 be super-polynomial recursive functions such that* $\mathrm{DTIME}(t_2(n)) - \mathrm{DTIME}(t_1(n)) \neq \emptyset$. *Then there exists a minimal pair in* $\mathrm{DTIME}(t_2(n)) - \mathrm{DTIME}(t_1(n))$.

Proof. Let $A \in \mathrm{DTIME}(t_2(n)) - \mathrm{DTIME}(t_1(n))$. Observe that for each recursive t, $\mathrm{DTIME}(t(n))$ is recursively presentable. Further, $\mathrm{DTIME}(t(n))$ is closed under finite variations if t is super-polynomial. Hence, apply the Minimal Pair Theorem with $A_1 = A_2 = A$ and $\mathscr{C} = \mathrm{DTIME}(t_1(n))$ and obtain a minimal pair $\{B_1, B_2\}$ such that $B_i \notin \mathrm{DTIME}(t_1(n))$ and $B_i \leq_m^P A$, $i = 1, 2$. Since the reduction f_i, $i = 1, 2$, from B_i to A can be seen to be linearly length-bounded, that is, $|f_i(x)| \leq |x|$, it follows that there exists a decision procedure for B_i of time complexity $p_i(|x|) + t_2(|f_i(x)|) \leq p_i(|x|) + t_2(|x|) \leq 2t_2(|x|)$, where p_i is a polynomial time bound for the computation of f_i. Hence, by the linear speed-up theorem, B_i, $i = 1, 2$, is in $\mathrm{DTIME}(t_2(n))$. □

Observe that a stronger interpretation of saying 'there exist arbitrarily complex minimal pairs' would be that for every (recursive) set A there exists a (recursive) set B such that $A \leq_T^P B$ and B is half of a minimal pair. Unfortunately, we cannot prove this by our methods.

Recursion theorists have considered the question of whether minimal pairs are low or high in the recursion theoretic sense [14]. It is possible to construct low minimal pairs as well as high minimal pairs (the latter requires infinite injury priority methods, cf. [14, p. 228]). Recently, a translation of the recursion theoretic low and high hierarchy into the P−NP-context has been achieved by this author [13]. For the exact definitions, please see [13]. By the previous applications of the Minimal Pair Theorem, it is easy to see that there exist minimal pairs in NP that are not in the low hierarchy in NP because the low hierarchy is recursively presentable [13]. On the other hand, if there is a low set in NP that is not in P (in the terminology of [13], $LH - P \neq \emptyset$) then there is also a low minimal pair in NP by Corollary 4. A sufficient condition for the existence of low sets in NP that are not in P is EXPTIME ≠ NEXPTIME, because by results in [5], all sets in NP over a one-letter alphabet are included in the second level of the low hierarchy in NP. We summarize this in the following corollary.

Corollary 7. *If* DEXPTIME ≠ NEXPTIME, *then there exists a minimal pair in* NP − P *over a one-letter alphabet.*

Proof. If DEXPTIME ≠ NEXPTIME, then by an easy padding argument (see [2]) there exist sets over a one-letter alphabet (tally sets) in NP − P. Let A_1 and A_2 be such a set, and let $\mathcal{C} = P$ in the Minimal Pair Theorem. Then the minimal pair provided by the theorem can easily be seen to be a tally set. □

5. Remarks

By a careful analysis the gap language can be seen to be recognizable in linear time. Such efficiency considerations can be found in [3].

Obviously, the proof of the Minimal Pair Theorem does not depend on the notion of $\leq_1^P$-reduction. It is possible to generalize the definition of minimal pair and the Minimal Pair Theorem to other subrecursive reducibilities defined in terms of restricting some kind of resource such as time or space by a recursive function. Also, it is possible to generalize the Minimal Pair Theorem to such reducibilities as γ- or R-reducibility (see [1]). Then the set F in the proof of the theorem is no longer in P but in NP ∩ co-NP or in R ∩ co-R (that is, the 0-degree of γ- or R-reducibility, respectively). A somewhat more general development can be found in [11].

Acknowledgment

The author wishes to thank Professor Ronald Book for his encouragement, José Balcázar for very useful discussions on the subject, the referees for their extremely constructive criticism, and Leslie Wilson for her careful typing of the manuscript.

References

[1] L. Adleman and K. Manders, Reducibility, randomness, and intractability, *Proc. 9th ACM Symp. Theory of Computing* (1977) pp. 151–163.
[2] R.V. Book, Tally languages and complexity classes, *Inform. and Control* **26** (1974) 186–193.
[3] S.I. Breidbart, The structure of complexity classes and degrees, Ph.D. Dissertation, University of California, Santa Barbara, 1977.
[4] P. Chew and M. Machtey, A note on structure and looking back applied to the relative complexity of computable functions, *J. Comput. System Sci.* **22** (1981) 53–59.
[5] K. Ko and U. Schöning, On circuit-size complexity and the low hierarchy in NP, *SIAM J. Comput.*, to appear.
[6] A.H. Lachlan, Bounding minimal pairs, *J. Symbolic Logic* **44** (1979) 626–642.
[7] A.H. Lachlan, Lower bounds for pairs of r.e. degrees, *Proc. London Math. Soc.* **16** (3) (1966) 537–569.
[8] R.E. Ladner, On the structure of polynomial time reducibility, *J. Assoc. Comput. Mach.* **22** (1975) 155–171.
[9] L.H. Landweber, R.J. Lipton and E.L. Robertson, On the structure of sets in NP and other complexity classes, *Theoret. Comput. Sci.* **15** (1981) 181–200.
[10] M. Machtey, Minimal pairs of polynomial degrees with subexponential complexity, *Theoret. Comput. Sci.* **2** (1976) 73–76.
[11] U. Schöning, Untersuchungen zur Struktur von NP und verwandten Komplexitätsklassen mit Hilfe verschiedener polynomieller Reduktionen, Dissertation, Univ. Stuttgart, 1981.
[12] U. Schöning, A uniform approach to obtain diagonal sets in complexity classes, *Theoret. Comput. Sci.* **18** (1982) 95–103.
[13] U. Schöning, A low and a high hierarchy within NP, *J. Comput. System Sci.* **27** (1983) 14–28.
[14] R. Soare, *Recursively Enumerable Sets and Degrees: The Study of Computable Functions and Computably Generated Sets* (Springer, Berlin, 1984) to appear.